Red Hat Network 4.0.5

Channel Management Guide

Red Hat Network 4.0.5: Channel Management Guide
Copyright © 2002 - 2005 Red Hat, Inc.

Red Hat, Inc.

1801 Varsity Drive
Raleigh NC 27606-2072 USA
Phone: +1 919 754 3700
Phone: 888 733 4281
Fax: +1 919 754 3701
PO Box 13588
Research Triangle Park NC 27709 USA

channel-mgmt(EN)-4.0.5-RHI (2005-06-30T16:20)
Copyright © 2005 by Red Hat, Inc. This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, V1.0 or later (the latest version is presently available at http://www.opencontent.org/openpub/).
Distribution of substantively modified versions of this document is prohibited without the explicit permission of the copyright holder.
Distribution of the work or derivative of the work in any standard (paper) book form for commercial purposes is prohibited unless prior permission is obtained from the copyright holder.
Red Hat and the Red Hat "Shadow Man" logo are registered trademarks of Red Hat, Inc. in the United States and other countries.
All other trademarks referenced herein are the property of their respective owners.
The GPG fingerprint of the security@redhat.com key is:
CA 20 86 86 2B D6 9D FC 65 F6 EC C4 21 91 80 CD DB 42 A6 0E

Table of Contents

1. Introduction **1**

2. Introduction to RHN Channels **3**

2.1. Base Channels and Child Channels 3
2.2. Subscribing to Channels 3
2.3. Channel Availability 4
2.4. Tools, Repositories, and Practices 4

3. Building Custom Packages **7**

3.1. Building packages for Red Hat Network 7
3.1.1. RPM Benefits 7
3.1.2. RHN RPM Guidelines 8
3.2. Digital Signatures for RHN Packages 10
3.2.1. Generating a GnuPG Keypair 10
3.2.2. Signing packages 12

4. Custom Channel and Package Management **13**

4.1. Channel Management Privileges 13
4.2. Manage Software Channels 14
4.3. Managed Software Channel Details 14
4.4. Manage Software Packages 16
4.5. Creating a Software Channel 17
4.6. Assigning Packages to Software Channels 18
4.7. Cloning Software Channels 18
4.8. Deleting Software Channels 19

5. Custom Errata Management **21**

5.1. Manage Errata 21
5.1.1. Published Errata 21
5.1.2. Unpublished Errata 21
5.2. Managed Errata Details 22
5.3. Creating and Editing Errata 22
5.4. Assigning Packages to Errata 23
5.5. Cloning Errata 24

6. Uploading and Maintaining Custom Packages **25**

6.1. Uploading Packages to RHN Proxy Server 25
6.1.1. Configuring and Using the **RHN Package Manager** 26
6.2. Uploading Packages to RHN Satellite Server 29
6.2.1. Configuring the **RHN Push** Application 29
6.2.2. Using the **RHN Push** application 32

Index **35**

Chapter 1.

Introduction

This document discusses issues surrounding the deployment and maintenance of customized software channels for RHN Proxy Server and RHN Satellite Server. It is used after the RHN Satellite Server or RHN Proxy Server is installed and configured. Refer to the *Red Hat Network 4.0.5 Satellite Installation Guide* and the *Red Hat Network 4.0 Proxy Installation Guide* for details, as well as the *Red Hat Network Client Configuration Guide*.

In some instances, this document refers to actions that are performed on the Red Hat Network Web servers. For RHN Proxy Server customers, this refers to the central Red Hat Network Servers at https://rhn.redhat.com. For Satellite customers, this refers to the RHN Satellite Server at your site.

Chapter 2.

Introduction to RHN Channels

A Red Hat Network channel is a collection of software packages. Channels help you segregate packages by sensible rules: a channel may contain packages from a specific Red Hat distribution, for instance. A channel may contain packages for an application or family of applications. Users may also define channels for their own particular needs; a company may create a channel that contains packages for all of the organization's laptops, for example.

2.1. Base Channels and Child Channels

There are two types of channels: *base channels* and *child channels*. A base channel consists of packages based on a specific architecture and Red Hat Enterprise Linux release. A child channel is a channel associated with a base channel that contains extra packages.

A system must be subscribed to only one base channel. A system can be subscribed to multiple child channels of its base channel. A subscribed system can only install or update packages available through its Red Hat Network channels.

When a system is registered with Red Hat Network, it is assigned to the base channel that corresponds to the system's version of Red Hat Enterprise Linux. Once a system is registered, its default base channel may be changed to a private base channel on a per-system basis via the RHN website. Alternately, you can have activation keys associated with a custom channel so that systems registering with those keys are automatically associated with the custom channel.

On the Red Hat Network website, the **Channels** page (located under the **Channels** tab on the top navigation bar) provides a list of all base channels and their child channels. Clicking on the name of a channel displays the **Channel Details** page, which provides a list of all of the packages in that channel, its errata, and any associated systems.

2.2. Subscribing to Channels

Subscribe systems to channels in the following ways:

- Registration through activation keys — Because of the simplicity and speed of activation keys, this is the preferred method for registering systems as clients of either RHN Proxy Server or RHN Satellite Server. Systems registered using an activation key are subscribed to all channels associated with that activation key. For more information on activation keys, consult the *Red Hat Network Client Configuration Guide* and the *Red Hat Network Reference Guide*.

- Install registration — When a system is initially registered through either the **Red Hat Update Agent** or the **Red Hat Network Registration Client**, it is automatically assigned to the base channel that corresponds to the version of Red Hat Enterprise Linux on the system. Once a system is registered, its default base channel may be changed to a private base channel on a per-system basis via the RHN website. Alternately, you can have activation keys associated with a custom channel so that systems registering with those keys are automatically associated with the custom channel. For more information on using these applications, refer to the respective chapter of the *RHN Reference Guide* for your entitlement level (Management or Provisioning).
- Website subscription — Various specific child channels are available for subscription, depending on the system's base channel. The system may be subscribed to the child channel through the RHN website. If you have created your own base channels, you may also reassign systems to these custom channels through the website. For more information on subscribing to channels online, refer to the Red Hat Network Website chapter of the *RHN Reference Guide*.

2.3. Channel Availability

There are many channels in Red Hat Network. Some are available to all users, some are available to users in a particular organization, and some are available only if you have purchased access to them. Channels fall into these main categories:

- Paid Service Channels — These channels are available if you who have purchased access to them either directly or in conjunction with a particular Red Hat solution. Red Hat Enterprise Linux is an example of a paid service channel.
- Custom Channels — You create these channels to manage custom packages. These channels, also known as *private channels*, appear only to the organization who creates them; they can never be accessed by anyone else.

This document focuses on the process of creating and maintaining custom channels with an RHN Proxy Server or on an RHN Satellite Server.

2.4. Tools, Repositories, and Practices

Before creating and managing channels, note the differences between the various tools and repositories at your disposal. This is especially important if you are deploying both an RHN Satellite Server and RHN Proxy Server, as this increases the utilities and storage locations available. Further, a Proxy-Satellite combination offers certain best practices for optimal performance.

First, become familiar with these package management tools:

- **RHN Package Manager** - Use this to push custom packages into custom channels on your RHN Proxy Server.
- **RHN Push** - Use this to push custom packages into custom channels on your RHN Satellite Server.
- **RHN Satellite Synchronization Tool** - Use this to import and synchronize standard packages from Red Hat Network to your RHN Satellite Server with Red Hat Network. This is done via the Internet or CD-ROM.

Each of these tools has a corresponding package repository. Both **RHN Package Manager** and **RHN Push** require the creation of a temporary staging directory for placement of custom packages that are uploaded to the Proxy or Satellite. You need to delete these staging directories after use.

Tip

Red Hat recommends archiving your custom packages externally from Red Hat Network.

If you are using both RHN Proxy Server and RHN Satellite Server, use only **RHN Push** and **RHN Satellite Synchronization Tool**. The Proxy-Satellite combination requires custom packages and channels be uploaded to the *Satellite only*. From there, the Proxy obtains the packages and distributes them to client systems.

Chapter 3.

Building Custom Packages

There are many things that might go wrong when building software packages. This is especially true when these packages must be delivered and installed through Red Hat Network. This chapter provides an overview of how to build packages for successful delivery via Red Hat Network. Topics covered include why to use RPM, how to build packages for RHN, and how to properly sign packages.

3.1. Building packages for Red Hat Network

Red Hat Network uses the *RPM Package Manager* (RPM) technology to determine what software additions and updates are applicable to each client system. Packages retrieved from Red Hat Network are usually in RPM format. Entire ISO images, however, are available through the **Software** tab of the Red Hat Network website, but are not available in RHN Satellite Server installations. If your Satellite has Solaris support enabled, you can use RHN Push to upload Solaris packages to custom channels used by Solaris clients.

RPM is a tool that provides users with a simple method for installing, uninstalling, upgrading, and verifying software packages. It also allows software developers to package the source code and compiled versions of a program for end users and developers.

3.1.1. RPM Benefits

RPM provides the following advantages:

Easy Upgrades

Using RPM, you upgrade individual components of a system without completely reinstalling. When Red Hat releases a new version of Red Hat Enterprise Linux, users do not have to reinstall in order to upgrade. RPM allows intelligent, fully-automated, in-place upgrades of your system. Configuration files in packages are preserved across upgrades so users do not lose customizations. There are no special upgrade files needed to update a package because the same RPM file is used to install and upgrade the package.

Package Querying

RPM provides querying options that allows you to search through your entire RPM database for all packages or just for certain files. You can also easily find out what package a file belongs to and from where the package came. The files contained in the package are in a compressed archive, with a custom binary header containing useful

information about the package and its contents. RPM queries the headers quickly and easily.

System Verification

Another feature is the ability to verify packages. If you are worried a file related to a package was deleted, you can verify the package to check the status of the files it provides. The verification notifies you of any anomalies. If errors do exist, you can re-install the files easily. Modified configuration files are preserved during reinstallation.

Pristine Sources

A crucial design goal of RPM is to allow the use of *pristine* software sources, as distributed by the original authors of the software. With RPM, the pristine sources can be packaged, along with any patches that were used, plus complete build instructions. This is an important advantage for several reasons. For instance, if a new version of a program is released, you do not necessarily have to start from scratch to make it compile. You can look at the patch to see what you *might* need to do. All the compiled-in defaults and changes made to get the software to build properly are easily visible using this technique.

Keeping sources pristine may seem important only to developers, but it results in higher quality software for end users, as well.

3.1.2. RHN RPM Guidelines

The strength of RPM lies in its ability to define dependencies and identify conflicts accurately. Red Hat Network relies on this aspect of RPM. Red Hat Network offers an automated environment, which means that no manual intervention can take place during the installation of a package. Therefore, when building RPMs for distribution through Red Hat Network, it is imperative to follow these rules:

1. Learn RPM. It is crucial to have a fundamental understanding of the important features of RPM to build packages properly. For more information about RPM, start with the following resources:
   - http://www.rpm.org/RPM-HOWTO/
   - http://www.redhat.com/docs/books/max-rpm/
   - http://www.rpm.org/mailing_list/
   - http://fedora.redhat.com/participate/developers-guide/ch-rpm-building.html

2. When building an RPM for a child channel, build the package on a fresh install of Red Hat Enterprise Linux of the same version as the child's base channel. Be sure to apply all updates from Red Hat Network first.

3. The RPM package must install without using the `--force` or `--nodeps` options. If you cannot install an RPM cleanly on your build system, Red Hat Network cannot install it automatically on a system.
4. The RPM package filename must be in the NVR (name, version, release) format and must contain the architecture for the package. The proper format is *`name-version-release.arch`*`.rpm`. For example, a valid RPM package filename is `pkgname-0.84-1.i386.rpm`, where name is *`pkgname`*, version is *`0.84`*, release is *`1`*, and arch is *`i386`*.
5. The RPM package should be signed by the maintainer of the package. Unsigned packages may be distributed through Red Hat Network, but the **Red Hat Update Agent** (`up2date`) must be forced to accept them. Signing packages is highly recommended and is covered in Section 3.2 *Digital Signatures for RHN Packages*.
6. If the package is changed in any way, including changing the signature or recompiling, the version or release must be increased incrementally. In other words, the NVRA (including architecture) for each RPM distributed through RHN must correspond to a unique build to avoid ambiguities.
7. No RPM package may obsolete itself.
8. If a package is split into separate packages, be extremely careful with the dependencies. Do not split an existing package unless there is a compelling reason to do so.
9. No package may rely upon interactive pre-install, post-install, pre-uninstall, or post-uninstall scripts. If the package requires direct user intervention during installation, it cannot work with Red Hat Network.
10. Any pre-install, post-install, pre-uninstall, and post-uninstall scripts should never write anything to stderr or stdout. Redirect the messages to `/dev/null` if they are not necessary. Otherwise, write them to a file.
11. When creating the spec file, use the group definitions from `/usr/share/doc/rpm-<`*`version`*`>/GROUPS`. If there is not an exact match, select the next best match.
12. Use the RPM dependency feature to make sure the program runs after it is installed.

Important

Do not create an RPM by archiving files and then unarchiving them in the post-install script. This defeats the purpose of RPM.

If the files in the archive are not included in the file list, they cannot be verified or examined for conflicts. In the vast majority of cases, RPM itself can pack and unpack archives most effectively anyway. For instance, don't create files in a `%post` that you don't clean up in a `%postun` section.

3.2. Digital Signatures for RHN Packages

All packages distributed through RHN should have a *digital signature*. A digital signature is created with a unique private key and can be verified with the corresponding public key. After creating a package, the SRPM (Source RPM) and the RPM can be digitally signed with a GnuPG key. Before the package is installed, the public key is used to verify the package was signed by a trusted party and the package has not changed since it was signed.

3.2.1. Generating a GnuPG Keypair

A GnuPG keypair consists of the private and public keys. To generate a keypair type the following command as the root user on the shell prompt:

```
gpg --gen-key
```

If you execute this command as a non-root user, you see the following message:

```
gpg: Warning: using insecure memory!
```

This message appears because non-root users cannot lock memory pages. Since you do not want anyone else to have your private GnuPG key or your passphrase, you want to generate the keypair as root. The root user can lock memory pages, which means the information is never written to disk.

After executing the command to generate a keypair, you see an introductory screen containing key options similar to the following:

```
gpg (GnuPG) 1.2.6; Copyright (C) 2004 Free Software Foundation, Inc.
This program comes with ABSOLUTELY NO WARRANTY.
This is free software, and you are welcome to redistribute it
under certain conditions. See the file COPYING for details.

Please select what kind of key you want:
   (1) DSA and ElGamal (default)
   (2) DSA (sign only)
   (4) RSA (sign only)
Your selection?
```

Accept the default option: `(1) DSA and ElGamal`. This option allows you to create a digital signature and encrypt/decrypt with two types of technologies. Type **1** and then press [Enter].

Next, choose the key size, which is how long the key should be. The longer the key, the more resistant against attacks your messages are. Creating a key of at least 1024 bits in size is recommended.

The next option asks you to specify how long you want your key to be valid. If you do choose an expiration date, remember that anyone who is using your public key must also

be informed of its expiration and supplied with a new public key. It is recommended that you select no expiration date. If you do not choose an expiration date, you are asked to confirm your decision:

```
Key does not expire at all
Is this correct (y/n)?
```

Press [y] to confirm your decision.

Your next task is to provide a User-ID containing your name, your email address, and an optional comment. Each is requested individually. When you are finished, you are presented with a summary of the information you entered.

Once you accept your choices, you enter a passphrase.

Tip

Like your account passwords, a good passphrase is essential for optimal security in GnuPG. Mix your passphrase with uppercase and lowercase letters, use numbers, and/or include punctuation marks.

Once you enter and verify your passphrase, your keys are generated. A message similar to the following appears:

```
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
+++++.+++++.++++++++....++++++++++..+++++.+++++.+++++++.+++++++
+++.++++++++++++++++++++++++++++++++++++++.........................++++
```

When the activity on the screen ceases, your new keys are placed in the directory `.gnupg` in root's home directory. This location is because you are ran the command as root. To list your root keys, use the command:

```
gpg --list-keys
```

The output is similar to the following:

```
/root/.gnupg/pubring.gpg
------------------------------------
pub  1024D/B7085C8A 2002-02-18 Your Name <you@example.com>
sub  1024g/E12AF9C4 2002-02-18
```

To retrieve your public key, use the following command:

```
gpg --export -a 'Your Name' > public_key.txt
```

Your public key is written to the file `public_key.txt`.

This public key is quite important. It's the key that must be deployed to all client systems that receive custom software through `up2date`. Techniques for deploying this key across an organization are covered in the *Red Hat Network Client Configuration Guide*.

3.2.2. Signing packages

Before you can sign packages, you need to configure your `~/.rpmmacros` file to include the following:

```
%_signature gpg
%_gpg_name B7085C8A
```

Replace the `_gpg_name` key ID value of *`B7085C8A`* with the key ID from your GPG keyring that you use to sign packages. This value tells **RPM** which signature to use.

To sign the package *`package-name-1.0-1.noarch.rpm`*, use the following command:

```
rpm --resign package-name-1.0-1.noarch.rpm
```

Enter your passphrase. To make sure the package is signed, use the following command:

```
rpm --checksig -v package-name-1.0-1.noarch.rpm
```

You should see the phrase `Good signature from "`*`Your Name`*`"` in the output, with *`Your Name`* replaced with the name associated with the signing key.

Chapter 4.

Custom Channel and Package Management

Custom channels allow administrators to use the Red Hat Network infrastructure to deploy packages built and maintained by their organizations. All channel and package management activities take place in the **Channels** tab of the RHN website. The instructions here are used in conjunction with the RHN website chapter of the *RHN Reference Guide*.

Tip

Because of the potential problems that may arise from deploying untested packages throughout your production environment, Red Hat strongly recommends creating beta channels covering select systems that can be used for staging.

For example, if you have a system group of Web servers that receives a set of custom packages, create temporary channels to install the packages on a non-critical subset of representative systems first. These might be development or staging servers, *not* live production systems. These temporary channels are then deleted using the steps described in Section 4.8 *Deleting Software Channels*.

4.1. Channel Management Privileges

In order to perform any channel management tasks, users must have obtained the proper permissions as a *Channel Administrator*. These permissions can be modified through the Red Hat Network website. Permissions are assigned to users by *Organization Administrators*, the highest level of administrator. Channel Administrator privileges are assigned as follows:

1. Log in to the Red Hat Network website as an Organization Administrator.
2. On the top navigation bar, click the **Users** tab and then click the name of the user who is performing channel management functions.
3. On the **User Details** page, scroll down to the **Roles** section and select the checkbox labeled **Channel Administrator**. Then click **Submit** at the bottom of the page. Note that Organization Administrators are automatically granted channel administration privileges.

4. Have the user log in to the Red Hat Network website, click the **Channels** tab on the top navigation bar, and ensure the **Manage Software Channels** button appears on the corresponding left navigation bar.

4.2. Manage Software Channels

In addition to the buttons and pages available to standard RHN Management-level users, RHN Satellite Server and RHN Proxy Server customers also have access to **Manage Software Channels** on the left navigation bar. This button opens the **Software Channel Management** interface, where all custom software channel management work occurs.

Warning

If you use both RHN Proxy Server and RHN Satellite Server, manage custom channels and packages *only* on the Satellite, since the Proxy servers receive updates directly from it. Manually managing packages and channels on a Proxy in this combined configuration risks putting your servers out-of-sync.

Clicking links within the **Software Channel Management** list takes you to different tabs of the **Managed Software Channel Details** page. Clicking a channel name opens the **Details** tab, while clicking its number of packages opens the **List/Remove** subtab of the **Packages** tab. Refer to Section 4.3 *Managed Software Channel Details* for a full explanation of these areas.

4.3. Managed Software Channel Details

Virtually all custom channel management tasks are carried out within the **Managed Software Channel Details** page, accessed by clicking **Manage Software Channels** on the left navigation bar and then selecting the name of channel to be altered. This page consists of two primary tabs: **Details** and **Packages**.

- **Details** — Provides basic information about the channel, such as its parent channel, name, summary, and description. Some of this information is modifiable. In addition, a **Globally Subscribable** checkbox can be seen by Organization Administrators and Channel Administrators. This signifies the default behavior of every channel allowing any user to subscribe systems to it. Unchecking this box and clicking **Update Channel** causes the appearance of a **Subscribers** tab, which is used to grant certain users subscription permissions to the channel.
- **Subscribers** — Presents a list of users who have subscription permissions to the custom channel. This tab appears when two conditions are true. First, the logged in user must be

an Organization Administrator or a Channel Administrator. Second, the **Globally Subscribable** checkbox on the **Details** tab must be unchecked, thereby making the channel subscribable by user. On this tab, select the checkboxes of the users to be allowed to subscribe systems to this channel and click **Update**. Note that Organization Administrators and Channel Administrators automatically have subscription access to all channels.

- **Managers** — Lists users who have management permissions to the custom channel. This tab appears for Organization Administrators and Channel Administrators. Select the checkboxes of the users to be allowed full administration of this channel and click **Update**. This status does not enable the user to create new channels. Note that Organization Administrators and Channel Administrators automatically have management access to all channels.
- **Errata** — Provides the errata associated with each of your custom channels. Just as Red Hat Network produces and delivers errata updates to Red Hat Enterprise Linux software, you deliver errata updates to your custom channels as part of updating your servers with the latest code. This tab contains subtabs that allow you to view, add, remove, and clone erratum: **List/Remove**, **Add** and **Clone**. Note that cloning errata can be done only via RHN Satellite Server.
  - **List/Remove** — Displays all of the errata currently associated with the custom channel and provides a means to cancel that association. To remove errata from the channel, select their checkboxes and click **Remove Errata** on the bottom right-hand corner of the page. A confirmation page appears listing the errata to be removed. Click **Confirm** to complete the action.
  - **Add** — Enables the addition of errata to the channel. All of the errata potentially applicable to the channel are listed. To add errata to the channel, select the appropriate checkboxes and click **Add Errata**. Refer to Chapter 5 *Custom Errata Management* for a discussion of errata management.
  - **Clone** — Allows Satellite customers to replicate errata and associated packages for a cloned channel. This subtab immediately appears populated for channels that were cloned with either the original state or select errata option. The **Clone** tab also gains errata whenever one is issued for the target (that is, originating) channel. This makes it useful for channels cloned with the current state option, as well. Refer to Section 4.7 *Cloning Software Channels* for a discussion of cloning options.

    To include errata from the target channel in the cloned channel, select either **Merge** or **Clone** from each advisory's dropdown menu. The **Merge** option exists only if the erratum has been previously cloned. Use it to associate the erratum across channels and avoid duplicate entries. Use the **Clone** option to create a new entry, such as when modifying it from the previous clone.

    By default, cloned errata inherit the label of the original Red Hat advisory with the "RH" prefix replaced with "CL". For example, RHSA-2003:324 becomes CLSA-2003:324. Subsequent clones of the same advisory have their second letters sequenced to denote their order, such as "CM" and "CN". These labels can be altered through the **Managed Errata Details** page. Refer to Section 5.2 *Managed Errata Details* for instructions.

In addition to the **Merge** option, previously cloned errata contain values within the **Owned Errata** column. The erratum label is linked to its details page. The **pub** and **mod** flags within parentheses identify whether the cloned erratum has been published or modified from the original advisory. A plus sign **+** before the flag indicates affirmative, the cloned errata has been published. A minus sign **-** before the flag denotes negative. For example, **(-mod)** may mean a package has been deleted. To find out more about publishing and editing custom errata, refer to Section 5.1 *Manage Errata*.

To exclude errata from the cloned channel, select **Do Nothing** from their dropdown menus. When satisfied with the changes, click **Clone Errata**. Review the impending changes on the confirmation page and click **Update Errata**.

- **Packages** — Provides the packages associated with each of your custom channels. This tab contains subtabs that allow you to view, add, and remove packages: **List/Remove**, **Add**, and **Compare**.
  - **List/Remove** — Displays all of the packages currently associated with the custom channel and provides a means to cancel that association. To remove packages from the channel, select their checkboxes and click **Remove Packages** on the bottom right-hand corner of the page. A confirmation page appears with the packages to be removed listed. Click **Confirm** to complete the action.

    Important

    This list differs from the package list available through the standard **Software Channel Details** page in that it displays all versions of a package remaining in the database, rather than just the latest. You may revert to a previous version of a package simply by removing the latest version.

  - **Add** — Enables the addition of packages to the channel. To see available packages, select an option from the **View** dropdown menu and click **View**. To add packages to the channel you are editing, select the appropriate checkboxes and click **Add Packages**. Refer to Section 4.6 *Assigning Packages to Software Channels* for a discussion of this process.
  - **Compare** — Enables the comparison of package lists between different channels. To see the differences, select another channel from the **Compare to:** dropdown menu and click **Compare**. A list appears showing all packages not contained by both channels and indicating the existing channel location of each.

4.4. Manage Software Packages

In addition to adding and removing packages within channels, you also have the option of deleting packages entirely from both the database and file system. Removal from the file system is delayed by about one hour. This can be done through the **Software Package Management** page, accessed by clicking **Manage Software Packages** on the left navigation bar.

Warning

Although deleting packages from the database can be undone by uploading them again, they lose their association with any errata. Upon reloading, they must be re-associated with errata manually. Refer to Chapter 5 *Custom Errata Management* for instructions.

To remove packages from the database, in the **Software Package Management** page, select an option containing them from the **View** dropdown menu and click **View**. Then select the appropriate checkboxes and click **Delete Packages**. A confirmation page appears with the packages listed. Click **Confirm** to delete the packages entirely.

Since the actual packages are stored on the RHN Proxy Server, its custom packages cannot be downloaded through the RHN website, although they are listed. They must be retrieved by the client system using `up2date`. Since the RHN Satellite Server provides its own website, its custom packages are accessible via HTTP or **Red Hat Update Agent**. To obtain custom packages, the client system must be subscribed to the channel containing the packages.

4.5. Creating a Software Channel

Before uploading packages to the server, a custom channel can be created to house them. Refer to Chapter 6 *Uploading and Maintaining Custom Packages* for instructions. Once uploaded, packages may be reassigned through the website, as described in Section 4.6 *Assigning Packages to Software Channels*.

Channels are created on the Red Hat Network website as follows:

1. Log in to the Red Hat Network website as a Channel Administrator.
2. On the top navigation bar, click the **Channels** tab and then click the **Manage Software Channels** button on the left navigation bar.
3. On the **Software Channel Management** page, click **create new software channel** at the top-right corner. RHN Satellite Server administrators are presented with the option to **clone channel**. Refer to Section 4.7 *Cloning Software Channels* for instructions.

4. On the **New Channel** page, define the details of the channel following the instructions on the page. For most channel management actions, the **Channel Label** is used to identify the channel, so select a meaningful label. View the details of existing channels for ideas.

   The **GPG key URL** must be the location of the key on the server, as defined during the client configuration process. Refer to the *Red Hat Network Client Configuration Guide*. The GPG key ID is the unique identifier, such as "DB42A60E", while the GPG key fingerprint is similar to "CA20 8686 2BD6 9DFC 65F6 ECC4 2191 80CD DB42 A60E". Notice that the key ID is the same as the last pair of quartets in the key fingerprint.

5. When finished, click **Create Channel** at the bottom of the page.

4.6. Assigning Packages to Software Channels

When packages are initially uploaded, they can be assigned to a custom channel, multiple custom channels, or no channel at all. Refer to Chapter 6 *Uploading and Maintaining Custom Packages* for instructions. Once uploaded, packages may be reassigned between custom channels and the No Channels repository.

These functions are made available by clicking the **Channels** tab in the top navigation bar and then **Manage Software Channels** on the left navigation bar. In the **Software Channel Management** page, click the name of the channel to receive packages.

In the **Managed Software Channel Details** page, click the **Packages** tab and then the **Add** subtab. To associate packages with the channel being edited, select the option now containing the packages from the **View** dropdown menu and click **View**. Packages already associated with the channel being edited are not displayed. Packages not assigned to a specific channel are found in the **Packages in no channels** menu item. Selecting **All managed packages** presents all available packages.

After clicking **View**, the package list for the selected option appears. Note that the page header still lists the channel being edited. In the package list, select the checkboxes of the packages to be assigned to the edited channel and click **Add Packages** at the bottom-right corner of the page. A confirmation page appears with the packages listed. Click **Confirm** to associate the packages with the channel. The **List/Remove** subtab of the **Managed Software Channel Details** page then appears with the new packages listed.

Once packages are assigned to a channel, the errata cache is updated to reflect the changes. This update is delayed briefly so that users may finish editing a channel before all of the changes are made available. To initiate your changes to the cache manually, click the **commit your changes immediately** link within the text at the top of the **List/Remove** subtab.

4.7. Cloning Software Channels

RHN Satellite Server Channel Administrators also have the ability to clone software channels for easy package association. Cloning offers you a complete replica of another channel, enabling you to immediately associate appropriate packages and errata with a custom software channel. To access this functionality, click the **Channels** tab on the top navigation bar, then the **Manage Software Channels** on the left navigation bar. This takes you to the **Software Channel Management** page. To begin cloning, click **clone channel** at the top-right corner.

You are immediately presented with three cloning options: current state of the channel, original state of the channel, or select errata. These options are described fully on the webpage itself but are summarized as:

- **Current state of the channel** — All of the errata and all of the latest packages now in the target channel.
- **Original state of the channel** — All of the original packages from the target channel but none of the errata or associated update packages.
- **Select Errata** — All of the original packages from the target channel with the ability to exclude certain errata and associated update packages.

Select the option you desire using the radio buttons within the **Clone** field, identify the target channel using the **Clone From** dropdown menu, and click **Create Channel**.

On the **New Software Channel** page, complete the fields as described in Section 4.5 *Creating a Software Channel*. The default values often suffice. When satisfied, click **Create Channel**. If you selected either the original or current option, you are directed to the **Details** tab of **Managed Software Channel Details** page, where you may alter settings for the new channel. Refer to Section 4.3 *Managed Software Channel Details* for instructions.

If you used the select errata option to clone the channel, you are instead directed to the **Clone** subtab of **Managed Software Channel Details** page, where you may individually select errata and associated packages for cloning and inclusion in the new channel. Refer to Section 4.3 *Managed Software Channel Details* for specific instructions.

4.8. Deleting Software Channels

RHN Satellite Server and RHN Proxy Server administrators also have the ability to remove unused channels. This action is conducted within the **Details** tab of the **Managed Software Channel Details** of a channel page. After opening up this tab, described in detail in Section 4.3 *Managed Software Channel Details*, click **delete software channel** at the top-right corner of the page to entirely remove the channel and all packages exclusively associated with it. Only the following page, click **Delete Channel** to finish the action.

Removing a channel via the website automatically deletes all packages associated only with that channel. Packages that are also associated with other channels are retained. If you have established that channel on a Proxy connected to a Satellite, you must delete the channel on the RHN Proxy Server.

Chapter 5.

Custom Errata Management

Custom errata enables you to issue errata alerts for the packages in your custom channels. All errata management activities take place in the **Errata** tab of the RHN website. The instructions here are used in conjunction with the RHN website chapter of the *Red Hat Network Reference Guide*.

5.1. Manage Errata

In addition to the buttons and pages available to standard RHN Management-level users, RHN Satellite Server and RHN Proxy Server customers also have access to **Manage Errata** in the left navigation bar. This button opens the **Errata Management** interface, where all custom errata management work occurs.

Warning

If you are using both RHN Proxy Server and RHN Satellite Server, you must manage errata only on the Satellite, since the Proxy servers receive updates directly from it. Managing errata on a Proxy in this combined configuration risks putting your servers out-of-sync.

Clicking on an advisory within the **Errata Management** list takes you to the **Details** tab of the **Managed Errata Details** page. Refer to Section 5.2 *Managed Errata Details* for a full explanation of this area.

5.1.1. Published Errata

The **Published Errata** page is shown by default when you click **Manage Errata** in the left navigation bar. It displays the errata alerts your organization has created and disseminated.

To edit an existing published errata, follow the steps described in Section 5.3 *Creating and Editing Errata*. To distribute the errata, click **Send Notification** on the top-right corner of the **Errata Details** page. The errata alert is sent to the administrators of all affected systems.

5.1.2. Unpublished Errata

The **Unpublished Errata** page appears when you click **Unpublished** below **Manage Errata** in the left navigation bar. It displays the errata alerts your organization has created but not yet distributed..

To edit an existing unpublished errata, follow the steps described in Section 5.3 *Creating and Editing Errata*. To publish the errata, click **Publish Errata** on the top-right corner of the **Errata Details** page. You then need to confirm the channels associated with the errata and click the **Publish Errata** button, now in the lower-right corner. The errata alert is shifted to the **Published** page awaiting distribution.

5.2. Managed Errata Details

If you click on the advisory of a managed errata alert in the **Published** or **Unpublished** pages, its **Managed Errata Details** page appears. This page is further divided into three tabs: **Details**, **Channels**, and **Packages**.

- **Details** — Provides the primary information you entered about the custom errata alert during its creation. This includes a synopsis, advisory name and type, related product, bugs, description, solution, keywords, references, and notes. To change any of this information, make your modifications in the appropriate fields and click **Update Errata**.
- **Channels** — Shows the channels associated with the selected errata. To change these associations, select or deselect the appropriate checkboxes and click the **Update Channels** button.
- **Packages** — Enables you to manage the packages associated with the selected errata. This tab contains two subtabs that allow you to view, add, and remove packages: **List/Remove** and **Add**.
  - **List/Remove** — Displays all of the packages currently associated with the custom errata and provides a means to cancel that association. To remove packages from the errata, select their checkboxes and click **Remove Packages** on the bottom right-hand corner of the page. A confirmation page appears listing the packages to be removed. Click **Confirm** to complete the action.
  - **Add** — Enables the addition of packages to the errata. To see available packages, select an option from the **View** dropdown menu and click **View**. To add packages to the errata you are editing, select the appropriate checkboxes and click **Add Packages**. Refer to Section 5.4 *Assigning Packages to Errata* for a comprehensive discussion of this process.

5.3. Creating and Editing Errata

Follow this procedure to make a custom errata alert.

1. On the top navigation bar click on **Errata** then click **Manage Errata** on the left navigation bar. From the **Errata Management** page, click on **create new erratum**.
2. Enter an intuitive label for the erratum in the **Advisory** field, ideally following a naming convention adopted by your organization. Note that this label cannot begin with the letters "RH" (capitalized or not) to prevent confusion between custom errata and those issued by Red Hat.
3. Then, complete all remaining required fields and click the **Create Errata** button. View standard Red Hat Errata Alerts for examples of properly completed fields.

RHN Satellite Server administrators may also create errata by cloning an existing one. This cloning preserves package associations and simplifies issuing errata. Refer to Section 5.5 *Cloning Errata* for instructions.

To edit an existing errata alert's details, click its advisory in the **Errata Management** page, make your changes in the appropriate fields of the **Details** tab, and click the **Update Errata** button. Click on the **Channels** tab to alter the errata's channel association. Click on the **Packages** tab to view and modify its packages.

To delete errata, select their checkboxes in the **Errata Management** page, click the **Delete Errata** button, and confirm the action. Note that deleting published errata may take a few minutes.

Tip

If you want to receive an email when errata alerts are issued for your systems, go to **Your RHN** => **Your Preferences** in the RHN Website and select **Receive email notifications**. This is a useful setting for administrators of subscribed systems in your organization.

5.4. Assigning Packages to Errata

Follow this procedure to assign packages to errata.

1. After selecting an erratum to edit, click on the **Packages** tab then the **Add** subtab.
2. To associate packages with the erratum being edited, select the channel from the **View** dropdown menu that contains the packages you want in and click **View**. Packages already associated with the erratum being edited are not displayed. Selecting **All managed packages** presents all available packages.

3. After clicking **View**, the package list for the selected option appears. Note that the page header still lists the errata being edited.
4. In the list, select the checkboxes of the packages to be assigned to the edited errata, and click **Add Packages** at the bottom-right corner of the page.
5. A confirmation page appears with the packages listed. Click **Confirm** to associate the packages with the errata. The **List/Remove** subtab of the **Managed Errata Details** page appears with the new packages listed.

Once packages are assigned to an erratum, the errata cache is updated to reflect the changes. This update is delayed briefly so that users may finish editing an erratum before all of the changes are made available. To initiate your changes to the cache manually, follow the directions to **commit your changes immediately** at the top of the page.

5.5. Cloning Errata

You may clone errata for easy replication and distribution as part of RHN Satellite Server. Only errata potentially applicable to one of your channels can be cloned. Errata can be applicable to a channel if that channel was cloned from a channel to which the errata applies. To access this functionality, click **Errata** on the top navigation bar, then **Clone Errata** on the left navigation bar. This button appears only for RHN Satellite Server customers.

Once in the **Clone Errata** page, select the channel containing the errata from the **View** dropdown menu and click **View**. Once the errata list appears, select the checkbox of the errata to be cloned and click **Clone Errata**. A confirmation page appears with the errata listed. Click **Confirm** to finish the cloning.

The cloned errata appears in your unpublished errata list. From there you can verify the errata text and the packages associated with that errata. Once you are ready, you can publish the errata so it is available to users in your organization.

Chapter 6.

Uploading and Maintaining Custom Packages

Depending upon which Red Hat Network service is used, there are two different mechanisms for uploading packages to private channels.

Customers of RHN Proxy Server use the **RHN Package Manager** application, which sends package header information to the central Red Hat Network Servers and places the package itself into the local repository of the Proxy that invoked **RHN Package Manager**.

Customers of RHN Satellite Server use the **RHN Push** application, which sends package header information to the local RHN Satellite Server and places the package into the local repository of the Satellite that invoked **RHN Push**.

This chapter discusses both of these tools in detail.

Warning

If you use both RHN Proxy Server and RHN Satellite Server, use only **RHN Push**. The Proxy-Satellite combination requires custom packages and channels be uploaded to the Satellite only. From there, the Proxy servers obtain the packages and distribute them to client systems.

6.1. Uploading Packages to RHN Proxy Server

RHN Package Manager allows an organization to serve custom packages associated with a private RHN channel through the RHN Proxy Server. If you want the RHN Proxy Server to serve only official Red Hat Enterprise Linux packages, you do not need to install **RHN Package Manager**.

To use the **RHN Package Manager**, install the `rhns-proxy-package-manager` RPM package and its dependencies. This package is available to registered RHN Proxy Server systems and is installed by running `up2date rhns-proxy-package-manager`.

Note

Only the header information for the packages is uploaded to the RHN Servers. The headers are required so that RHN can resolve package dependencies for the client systems. The actual package files (`*.rpm`) are stored on the RHN Proxy Server. For this reason,

custom packages cannot be downloaded through the RHN website, although they are listed. They must be retrieved by the client system using `up2date`.

6.1.1. Configuring and Using the RHN Package Manager

Before you can use RHN Package Manager to upload packages into RHN, you need to first manually copy the packages to the Proxy server itself. For example, from a development host you can use `scp`:

```
scp foo.rpm root@rhnproxy.example.com:/tmp
```

When using RHN Package Manager to upload the packages into Red Hat Network, point at the files you previously copied to the server.

Tip

Create at least one private channel to receive custom packages prior to upload into Red Hat Network, since a channel is required for systems to obtain the packages.

The following command uploads the package headers to the RHN Servers and copies the packages to the RHN Proxy Server repository:

```
rhn_package_manager -c label_of_private_channel pkg-list
```

The *`label_of_private_channel`* is the custom channel created to receive these packages. Be sure you use the precise channel label specified during its creation. If you have one or more channels specified (using `-c` or `--channel`), the uploaded package headers are linked to all the channels identified. If you do not specify a channel, the packages are deposited in the **No Channels** section of the **Package Management** page. Refer to Section 4.6 *Assigning Packages to Software Channels* for instructions on reassigning packages.

The *`pkg-list`* reference represents the list of packages to be uploaded. These packages must already be physically copied to the Proxy host. Alternatively, use the `-d` option to specify the local directory that contains the packages to be added to the channel. **RHN Package Manager** can also read the list of packages from standard input (using `--stdin`).

Other options are specified in a configuration file, such as the Red Hat Network Server URL, the HTTP proxy username and password (if your HTTP proxy requires authentication), and the top directory where packages live. This special configuration must *not be edited* and is located at `/etc/rhn/default/rhn_proxy_package_manager.conf`. You can override the choices in that default configuration file with settings in the main con-

figuration file `/etc/rhn/rhn.conf` or via command line options passed to RHN Package Manager.

Parameters not set in this file are read from `.rhn_package_manager` in the home directory of the user currently logged in and finally from `/etc/rhn/rhn_package_manager.conf`. Make sure all of these files have the appropriate permissions to prevent others from reading them.

After uploading the packages, check to see if the local directory is in sync with the RHN Server's image of the channels:

```
rhn_package_manager -s -c name_of_private_channel
```

This `-s` option lists all the missing packages, which are packages uploaded to the RHN Server but not present in the local directory. You must be an Organization Administrator to use this option. The application prompts you for your RHN username and password.

The `--copyonly` option copies the file listed in the argument into the specified channel without uploading to the Satellite. This is useful when a channel on a RHN Proxy Server is missing a package and you don't want to reimport all of the packages in the channel.

```
rhn_package_manager -c channel-name --copyonly /path/to/missing/file
```

You can also use **RHN Package Manager** to retrieve a list of packages in a channel, as they are stored by the RHN Server:

```
rhn_package_manager -l -c name_of_private_channel
```

The `-l` option lists the package name, version number, release number, architecture, and channel name for each package in the specified channel(s). Refer to Table 6-1 for additional options.

Table 6-1 is a summary of all the command line options for **RHN Package Manager** (`rhn_package_manager`):

Option	Description
`-v, --verbose`	Increase verbosity of standard output messages.
`-d, --dir` *DIRECTORY_NAME*	Process packages from this directory.
`-c, --channel` *CHANNEL_NAME*	Specify the channel to receive packages. Multiple channels may be specified using multiple instances of `-c` (e.g.: `-c channel_one -c channel_two`)
`-n, --count` *NUMBER*	Process this number of headers per call — the default is 32.

Option	Description
`-l, --list`	List the packages in the specified channel(s).
`-s, --sync`	Check if local directory is in sync with the server.
`-p, --printconf`	Print the current configuration and exit.
`--newest`	Push only the packages that are newer than those on the server. Note that source packages are special in that their versions are never compared to each other. Their newness is dependent on their associated binary packages. Using this option with RHN Package Manager and just a source package does upload the package, but the source package does not appear in the RHN Web interface until the associated binary package has been uploaded. Contrast this with `--source`. Using `--source --newest` together *does* update the stand-alone source package with newer packages and does not require an associated binary package to be uploaded first.
`--source`	Upload the indicated source packages. Doing this treats them as plain, stand-alone packages and *not* as special source packages associated with another, pre-existing binary package. For example, you can use this when you want to distribute application source to developers and testers outside of regular source control management.
`--stdin`	Read the package names from standard input.
`--nosig`	Don't fail if packages are unsigned.
`--no-ssl`	Turn off SSL (not recommended).
`--stdin`	Read the package names from standard input.
`--username` *USERNAME*	Specify RHN username. If not provided, you are prompted for the username of a valid Channel Administrator.
`--password` *PASSWORD*	Specify RHN password. If not provided, you are prompted for the password of a valid Channel Administrator.
`--dontcopy`	In the post-upload step, do not copy the packages to their final location in the package tree.
`--copyonly`	Only copy the packages, do not re-import them.

Option	Description
`--test`	Only print a list of the packages to be pushed.
`-?, --help`	Display the help screen with a list of options.
`--usage`	Briefly describe the available options.
`--copyonly`	Only copy packages

Table 6-1. `rhn_package_manager` options

Tip

These command line options are also described in the `rhn_package_manager` manual page: `man rhn_package_manager`.

6.2. Uploading Packages to RHN Satellite Server

The **RHN Push** application allows you to serve custom packages associated with a private RHN channel through the RHN Satellite Server. If you want the RHN Satellite Server to serve only official Red Hat Enterprise Linux packages, you do not need to install **RHN Push**.

To use **RHN Push**, install the `rhnpush` package and its dependencies. This package is available to registered RHN Satellite Server systems and is installed by running `up2date rhnpush`.

RHN Push uploads RPM header information to the RHN Satellite Server database and places the RPM in the RHN Satellite Server package repository. Unlike the RHN Proxy Server's **RHN Package Manager**, **RHN Push** never distributes package information, even the headers, beyond the RHN Satellite Server database.

Tip

If your Satellite installation is enabled to support Solaris OS systems, you may use RHN Push from a Solaris client to upload Solaris package content to custom Solaris channels.

6.2.1. Configuring the RHN Push Application

When **RHN Push** is installed, a central configuration file is installed in `/etc/sysconfig/rhn/rhnpushrc`. This file contains values for all the options contained in Table 6-2. In addition, the application creates `~/.rhnpushrc` and `./.rhnpushrc` upon its initial use.

These distinct configuration files are useful in varying your settings depending on the directory from which the `rhnpush` command is issued. Settings in the current directory (`./.rhnpushrc`) take precedent over those in the user's home directory (`~/.rhnpushrc`), which are used before those in the central configuration file (`/etc/sysconfig/rhn/rhnpushrc`).

For instance, you can use the current directory configuration file to specify the software channel to be populated, the home directory configuration file to include the username to be invoked, and the central configuration file to identify the server to receive the packages.

Table 6-2 contains all command line options for the `rhnpush` command:

Option	Description
`-v --verbose`	Increase verbosity, option can be used multiple times, that is, `-vv`, `-vvv`, and so forth.
`-d, --dir` *DIRECTORY*	Process packages from this directory.
`-c, --channel` *CHANNEL_LABEL*	Specify the channel to receive packages. Note that this is required and is not the same as the channel's name. Multiple channels may be specified using multiple instances of `-c` (e.g. `-c=CHANNEL_ONE -c=CHANNEL_TWO`).
`-n, --count` *N_HEADERS_PER_CALL*	Process this number of headers per call. Must be an integer. The default number is 25.
`-l, --list`	List only the specified channels.
`-r, --reldir`*RELATIVE_DIRECTORY*	Associate this relative directory with each file.
`-o, --orgid`*ORGANIZATION_ID*	Include your organization's ID number. Must be an integer.

Option	Description
`-u , --username` *USERNAME*	Include the RHN username of the user that has administrative access to the specified channel. If not provided, `rhnpush` prompts for the username of a valid Channel Administrator. The username and password are cached in `~/.rhnpushcache` for a limited time, five minutes being the default. Use `--new-cache` to force a new username and password.
`-p , --password` *PASSWORD*	Include RHN password of user that has administrative access to the specified channel. If not provided, `rhnpush` prompts for the password of a valid Channel Administrator. The username and password are cached in `~/.rhnpushcache` for a limited time, five minutes being the default. Use `--new-cache` to force a new username and password.
`-s, --stdin`	Read package list from standard input, for example from a piped `ls` command.
`-X, --exclude` *GLOB*	Exclude packages that match this glob expression.
`--force`	Force upload of a package, even if a package of that name and version currently exists in the channel. Without this option, uploading a pre-existing package returns an error.
`--nosig`	Don't fail if packages are unsigned.
`--new-cache`	Forces RHN Push to drop the username and password cache, then accept or ask for new ones. This is useful if you make a mistake entering them the first time.

Option	Description
`--newest`	Push only the packages that are newer than those on the server. Note that source packages are special in that their versions are never compared to each other. Their newness is dependent on their associated binary packages. Using this option with RHN Push and just a source package does upload the package, but the source package does not appear in the RHN Web interface until the associated binary package has been uploaded. Contrast this with `--source`. Using `--source --newest` together *does* update the stand-alone source package with newer packages and does not require an associated binary package to be uploaded first.
`--header`	Upload only the headers.
`--source`	Upload the indicated source packages. Doing this treats them as plain, stand-alone packages and *not* as special source packages associated with another, pre-existing binary package. For example, you can use this when you want to distribute application source to developers and testers outside of regular source control management.
`--server` *SERVER*	Specify the server to which packages are uploaded. Currently, a value of `http://localhost/APP` is necessary. This parameter is required.
`--test`	Only print a list of the packages to be pushed, don't actually push them.
`-h, --help`	Briefly describe the options.
`-?, --usage`	View the usage summary.

Table 6-2. `rhnpush` options

Tip

These command line options are also described in the `rhnpush` manual page: `man rhnpush`.

6.2.2. Using the RHN Push application

Note

It is recommended that you create at least one private channel to receive custom packages prior to upload, since a channel is required for systems to obtain the packages.

The following command uploads package headers to the RHN Satellite Server and copies the packages to the RHN Satellite Server package repository:

```
rhnpush -c label_of_private_channel pkg-list
```

You can override the settings in your **RHN Push** configuration file(s) by specifying options and values on the command line:

```
rhnpush -c label_of_private_channel --server localhost pkg-list
```

The *label_of_private_channel* is the custom channel created to receive these packages. Be sure you use the precise channel label specified during its creation. If you have one or more channels specified (using `-c` or `--channel`), the uploaded package headers are linked to all the channels identified. If you do not specify a channel, the packages are deposited in the **No Channels** section of the **Package Management** page. Refer to Section 4.6 *Assigning Packages to Software Channels* for instructions on reassigning packages.

The `--server` option specifies the server to which the packages are installed, and is required. **RHN Push** can be installed on external systems, but running **RHN Push** locally on the RHN Satellite Server is recommended.

The *pkg-list* reference represents the list of packages to be uploaded. Alternatively, use the `-d` option to specify the local directory that contains the packages to be added to the channel. **RHN Push** can also read the list of packages from standard input (using `--stdin`).

Index

upload packages to RHN Proxy Server, 25

C

Channels
  cloning, 19
  deleting, 19
  intro, 3
custom packages, 7
  building, 7
    guidelines, 8
  signing, 12
  upload to RHN Proxy Server, 25
  upload to RHN Satellite Server, 29

E

errata alerts
  cloning, 24
  creating and editing, 23
  managing, 21
  managing published, 21
  managing unpublished, 22

G

GnuPG key
  creating, 10
  signing packages with, 12
gpg key, 10

H

how to
  build custom packages, 7
  clone a channel, 19
  configure RHN Package Manager, 26
  configure RHN Push, 30
  copy missing packages to Satellite, 27
  deliver non-RPM packages, 29
  generate a GnuPG key, 10
  retrieve channel package list, 27

M

Manage Errata
  viewing details, 22
Managed Channel Details, 14
managed software channels
  details, 14

P

packages
  Solaris and UNIX, 29

R

RHN Package Manager, 25
  channels, specifying, 26
  configuration file, 26
  configuring, 26
  copy missing packages to Satellite, 27
  installing, 25
  retrieve channel package list, 27
  rhn_package_manager, 27
  upload package headers, 26
  verify local package list, 27
RHN Push
  channels, specifying, 33
  configuring, 30
  installing, 29
  using, 33
rhn_package_manager, 26
    (see also RHN Package Manager)
  command line options, 27
rhn_package_manager.conf, 26
RPM
  benefits, 7
RPM Package Manager
  (see RPM)

S

Software
    Channel Management, 14

U

upload packages, 25

W

website
    Manage Software Channels, 14
what are
    benefits of RPM, 7